

Message Encoding and Retrieval for Spread and Cyclic Orbit Codes

Anna-Lena Horlemann-Trautmann
 Algorithmics Laboratory
 EPF Lausanne, Switzerland
 Email: anna-lena.horlemann@epfl.ch

Abstract

Spread codes and cyclic orbit codes are special families of constant dimension subspace codes. These codes have been well-studied for their error correction capability, transmission rate and decoding methods, but the question of how to encode and retrieve messages has not been investigated. In this work we show how a message set of consecutive integers can be encoded and retrieved for these two code families.

Index Terms

Message encoding, network coding, constant dimension codes, subspace codes, Grassmannian, enumerative coding, orbit codes, finite spreads, discrete logarithm.

I. INTRODUCTION

Random network coding has received much attention in the last decade. *Subspace codes*, first introduced in [21], are a class of codes used for random network coding. They are defined to be sets of subspaces of some given ambient space \mathbb{F}_q^n of dimension n over the finite field \mathbb{F}_q with q elements. When we restrict ourselves to subspace codes, whose codewords all have the same dimension k , we talk about *constant dimension codes*.

A closely related area of research are rank-metric codes. These codes have already been studied before subspace codes, and it is known that one can construct optimal rank-metric codes, called maximum rank distance (MRD) codes, for any set of parameters. In [5], [10] a general construction for MRD codes was given. These codes are also called Gabidulin codes and they can be represented as a linear block code over some extension field of the underlying field.

Since the main idea of coding theory is to transmit information through a communication channel, any code should be able to encode information, or, in other words, *encode messages* from a given message set. For generality, we will choose as message set the first non-negative integers $\mathcal{M} = \{0, 1, 2, \dots\}$. A message encoding map is an injective map from \mathcal{M} to the code. The corresponding *message retrieval* map is the inverse of the encoding map. From an application point of view it is very important that a code has an efficiently computable message encoding and corresponding retrieval map, since these need to be computed for every information transmission.

In the seminal paper [21] a class of Reed-Solomon-like constant dimension codes is proposed, which was later on shown to be equivalent to the lifting of Gabidulin codes [32]. Due to the linearity of the Gabidulin code over an extension field, there exist efficient message encoding and retrieval maps for these codes, in analogy to the encoding and retrieval maps of linear block codes.

During the last years other constructions of subspace codes were developed, e.g. in [3], [6], [7], [8], [11], [12], [15], [17], [20], [23], [31], [33], [37], [39]. Some of these constructions have the mere purpose of giving an improved transmission rate (i.e., larger cardinality of the code for the same parameters), while others have algebraic structure that can be used e.g. for decoding. The constructions from [6], [31] are based on the Reed-Solomon-like construction from [21], hence message encoding and retrieval can still be done based on the linearity of the underlying Gabidulin codes.

However, for most of the other known subspace code constructions, the corresponding codes cannot be represented as a linear block code over some extension field, and it is not obvious how message encoding and message retrieval can be done for these codes. Surprisingly, this problem of message encoding and retrieval has received little attention in the before-mentioned and other related papers and will be the topic of this paper. We want to study this problem for two classes of subspace codes, namely *spread codes* [17], [23] and *orbit codes* [15], [37], [39]. These two classes are of particular interest, since spread codes are optimal (see e.g. [34]) with respect to their rate for a given error correction capability (and thus achieve a better rate than the codes from [21]), and orbit codes have a lot of structure, which gives rise to code constructions and efficient error correcting decoding algorithms (see e.g. [37]).

The paper is organized as follows: In the following section we will give some preliminaries about finite fields and subspace codes, among others the definitions and constructions of spread codes and orbit codes. In Section III we derive some preliminary results on computational complexities of tasks that we need later on in our message encoding and retrieval algorithms. In Section

IV we derive an efficient encoding map for Desarguesian spread codes. In Section V we investigate message encoding for cyclic orbit codes. In Section VI we briefly describe how a message encoding and retrieval algorithm for a given subspace code can be combined with an error correcting decoding algorithm for another, semi-linearly isometric, code. We conclude this work in Section VII.

II. PRELIMINARIES

In this section we give all the preliminaries we will need later on in the paper. We will first introduce finite fields and recall known results related to finite fields. Then we will do the same for subspace codes, where we also define spread and orbit codes. In the third subsection we define message encoding and retrieval maps and give a short overview of known results.

A. Finite Fields

In this subsection we recall some known facts about finite fields. The definitions and results can be found in any textbook on finite fields, e.g. in [22].

Let q be a prime power. We denote the finite field with q elements by \mathbb{F}_q .

Definition 1. A polynomial $p(x) \in \mathbb{F}_q[x]$ is called *irreducible*, if it cannot be factored into the product of two non-constant polynomials of $\mathbb{F}_q[x]$.

Lemma 2. 1) If $\deg(p(x)) = k$, $p(x)$ is irreducible and α is a root of $p(x)$, then $\mathbb{F}_{q^k} \cong \mathbb{F}_q[\alpha]$.
2) If furthermore $\text{ord}(\alpha) = q^k - 1$, we say that α and $p(x)$ are primitive. In this case, $\mathbb{F}_{q^k}^* \cong \langle \alpha \rangle$.

Lemma 3. The following map ψ_k is a vector space isomorphism between \mathbb{F}_q^k and $\mathbb{F}_q[\alpha]$:

$$\begin{aligned} \psi_k : \mathbb{F}_q^k &\longrightarrow \mathbb{F}_q[\alpha] \\ (u_1, \dots, u_k) &\longmapsto \sum_{j=1}^k u_j \alpha^{j-1} \end{aligned}$$

Definition 4. The companion matrix P of some monic polynomial $p(x) = \sum_{i=0}^k p_i x^i$ is defined as

$$P = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & 0 & \dots & 1 \\ -p_0 & -p_1 & -p_2 & \dots & -p_{k-1} \end{pmatrix}.$$

Note that one often finds the transpose definition of a companion matrix in the literature. However, in this work we will use the above row-wise definition. One can verify that multiplication with P , respectively α , commutes with the vector space isomorphism ψ_k , which is stated in the following lemma.

Lemma 5. Let $p(x) \in \mathbb{F}_q[x]$ be monic, irreducible of degree k . Moreover, let α be a root of $p(x)$ and $P \in \mathbb{F}_q^{k \times k}$ the corresponding companion matrix. Then

$$\psi_k(u)\alpha = \psi_k(uP)$$

for any $u \in \mathbb{F}_q^k$. This implies that

$$\mathbb{F}_q[\alpha] \cong \mathbb{F}_q[P].$$

Throughout the paper we will denote by $\rho : \mathbb{F}_q[\alpha] \rightarrow \mathbb{F}_q[P]$ the isomorphism given by $\rho(\alpha^i) = P^i$ and $\rho(0) = 0_{k \times k}$.

To set up our message encoding and retrieval maps later on, we need the following bijections between vector spaces over finite fields and integers sets.

Definition 6. Let p be prime and m any positive integer. The map

$$\begin{aligned} \phi'_m : \mathbb{F}_p^m &\longrightarrow \{0, 1, \dots, p^m - 1\} \\ (u_1, \dots, u_m) &\longmapsto \sum_{i=1}^m u_i p^{i-1} \end{aligned}$$

is called the inverse p -adic expansion.

It is well-known that, for a prime number p , ϕ'_m is a bijection. This map can be extended to a q -adic expansion, for a prime power $q = p^r$, by fixing a bijection φ' between $\mathbb{F}_q = \mathbb{F}_{p^r}$ and $\{0, \dots, q-1\}$. To do so we represent $\mathbb{F}_{p^r} \cong \mathbb{F}_p[\beta]$ for a suitable β , and choose the bijection

$$\begin{aligned} \varphi' : \mathbb{F}_p[\beta] &\longrightarrow \{0, 1, \dots, q-1\} \\ \sum_{i=1}^r u_i \beta^{i-1} &\longmapsto \sum_{i=1}^r u_i p^{i-1}. \end{aligned}$$

One can easily see that $\varphi'(1) = 1$ and $\varphi'(0) = 0$.

Definition 7. The inverse q -adic expansion is given by

$$\begin{aligned} \phi''_m : \mathbb{F}_q^m &\longrightarrow \{0, 1, \dots, q^m - 1\} \\ (u_1, \dots, u_m) &\longmapsto \sum_{i=1}^m \varphi'(u_i) q^{i-1}. \end{aligned}$$

To furthermore extend this map to a q^k -adic expansion, we need to fix a bijection φ between \mathbb{F}_{q^k} and $\{0, \dots, q^k - 1\}$. To do so we represent $\mathbb{F}_{q^k} \cong \mathbb{F}_q[\alpha]$ for some suitable α and choose the bijection

$$\begin{aligned} \varphi : \mathbb{F}_q[\alpha] &\longrightarrow \{0, 1, \dots, q^k - 1\} \\ \sum_{i=1}^k u_i \alpha^{i-1} &\longmapsto \sum_{i=1}^k \varphi'(u_i) q^{i-1}. \end{aligned}$$

One can again easily see that $\varphi(1) = 1$ and $\varphi(0) = 0$.

Definition 8. The inverse q^k -adic expansion is given by

$$\begin{aligned} \phi_{k,m} : \mathbb{F}_{q^k}^m &\longrightarrow \{0, 1, \dots, q^{km} - 1\} \\ (u_1, \dots, u_m) &\longmapsto \sum_{i=1}^m \varphi(u_i) q^{k(i-1)}. \end{aligned}$$

It can easily be verified that $\phi_{k,m}$ is again a bijection. For computing the preimage $(u_1, \dots, u_m) \in \mathbb{F}_{q^k}^m$ of some $j \in \{0, \dots, q^{km} - 1\}$, i.e., the q^k -adic expansion of j , one recursively computes $\varphi(u_{\ell+1}) \equiv (j - \sum_{i=1}^{\ell} \varphi(u_i) q^{k(i-1)}) / q^{\ell k} \pmod{q^k}$ with the initial congruence $\varphi(u_1) \equiv j \pmod{q^k}$.

B. Subspace Codes

We denote the set of all subspaces of \mathbb{F}_q^n by $\mathcal{P}_q(n)$ and the set of all subspaces of \mathbb{F}_q^n of dimension k , called the *Grassmannian*, by $\mathcal{G}_q(k, n)$. We represent a vector space $\mathcal{U} \in \mathcal{G}_q(k, n)$ by a matrix $U \in \mathbb{F}_q^{k \times n}$ such that the row space of U , denoted by $\text{rs}(U)$, is equal to \mathcal{U} .

Definition 9. A *subspace code* is simply a subset of $\mathcal{P}_q(n)$ and a *constant dimension code* is a subset of $\mathcal{G}_q(k, n)$.

The following is a metric on $\mathcal{P}_q(n)$, and hence also on $\mathcal{G}_q(k, n)$ (see e.g. [21]).

Definition 10. The *subspace distance* is defined as

$$d_S(\mathcal{U}, \mathcal{V}) := \dim(\mathcal{U}) + \dim(\mathcal{V}) - 2 \dim(\mathcal{U} \cap \mathcal{V})$$

for any $\mathcal{U}, \mathcal{V} \in \mathcal{P}_q(n)$.

The minimum distance $d_S(\mathcal{C})$ of a subspace code $\mathcal{C} \subseteq \mathcal{P}_q(n)$ is defined in the usual way, as the minimum of all pairwise distances of the codewords, i.e.,

$$d_S(\mathcal{C}) := \min\{d_S(\mathcal{U}, \mathcal{V}) \mid \mathcal{U}, \mathcal{V} \in \mathcal{C}, \mathcal{U} \neq \mathcal{V}\}.$$

Since the dual of a subspace code \mathcal{C} has the same minimum distance as \mathcal{C} (see e.g. [21]), it is customary to restrict oneself to $k \leq n/2$, which we will assume throughout the paper.

We will now introduce spread and orbit codes. These families of constant dimension codes will be the focus of this paper.

Definition 11. A *spread*, in $\mathcal{G}_q(k, n)$ is defined as a set of elements of $\mathcal{G}_q(k, n)$ that pairwise intersect only trivially and cover the whole space \mathbb{F}_q^n .

Spreads are well-known geometrical objects, see e.g. [18]. Since spreads are subsets of $\mathcal{G}_q(k, n)$, they can be used as constant dimension codes. In this case one also speaks of *spread codes*, see e.g. [23]. The following properties of spread codes are well-known and can easily be derived.

Lemma 12. [18], [23]

- 1) *Spreads in $\mathcal{G}_q(k, n)$ exist if and only if $k|n$.*
- 2) *A spread in $\mathcal{G}_q(k, n)$ has minimum subspace distance $2k$ and cardinality $(q^n - 1)/(q^k - 1)$.*
- 3) *A constant dimension code in $\mathcal{G}_q(k, n)$ with minimum subspace distance $2k$ and cardinality $(q^n - 1)/(q^k - 1)$ is a spread.*

For more information on different constructions and decoding algorithms of spread codes, see [16], [23], [24], [34]. We will use the following well-known construction, which gives rise to a *Desarguesian spread code* in $\mathcal{G}_q(k, n)$ [1], [34].

Construction I:

- Let α be a root of an irreducible polynomial $p(x) \in \mathbb{F}_q[x]$ of degree k and let P be the corresponding companion matrix. Denote by $\rho : \mathbb{F}_q[\alpha] \rightarrow \mathbb{F}_q[P]$ the isomorphism from the previous subsection.
- Represent \mathbb{F}_{q^k} as $\mathbb{F}_q[\alpha]$. Let $m := n/k$ and consider $\mathcal{G}_{q^k}(1, m)$, which has $q^{k(m-1)} + q^{k(m-2)} + q^{k(m-3)} + \dots + 1 = (q^n - 1)/(q^k - 1)$ elements. Naturally, all these lines intersect only trivially.
- Define the map

$$\begin{aligned} \text{des} : \quad \mathcal{G}_{q^k}(1, m) &\longrightarrow \mathcal{G}_q(k, n) \\ \text{rs}(v_1, v_2, \dots, v_m) &\longmapsto \text{rs}(\rho(v_1), \rho(v_2), \dots, \rho(v_m)). \end{aligned}$$

Then the image of des is a Desarguesian spread in $\mathcal{G}_q(k, n)$.

Note that the name *Desarguesian* arises from the fact that the translation planes of these spreads are Desarguesian planes. For our purposes though, this fact is not needed, we simply use the name for the construction from above.

In our message encoding algorithms for spread codes we need to have a unique description of the elements of $\mathcal{G}_{q^k}(1, m)$. To do so we choose the normalized basis vector, i.e., the one element of the one-dimensional subspace whose first non-zero entry is equal to one. This normalized vector is then mapped by des to the reduced row echelon form of the respective codeword. The reader familiar with projective spaces will notice that $\mathcal{G}_{q^k}(1, m)$ corresponds exactly to the projective space over \mathbb{F}_{q^k} of dimension $m - 1$. The usage of normalized representatives of points is a common concept there.

Example 13. Let $q = k = m = 2, n = 4$ and α be a root of $x^2 + x + 1$, i.e., a primitive element of $\mathbb{F}_{2^2} \cong \mathbb{F}_2[\alpha]$. The respective companion matrix is

$$P = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

Then $\mathcal{G}_{2^2}(1, 2) = \{\text{rs}(1, 0), \text{rs}(1, \alpha), \text{rs}(1, \alpha^2), \text{rs}(1, 1), \text{rs}(0, 1)\}$ and substituting all elements of $\mathbb{F}_2[\alpha]$ with its corresponding element from $\mathbb{F}_2[P]$ gives a spread in $\mathcal{G}_2(2, 4)$:

$$\left\{ \text{rs} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \text{rs} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}, \text{rs} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \text{rs} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \text{rs} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \right\}$$

Orbit codes [38] in $\mathcal{G}_q(k, n)$ are defined to be orbits of a subgroup of the general linear group $\text{GL}_n := \{A \in \mathbb{F}_q^{n \times n} \mid \text{rank}(A) = n\}$ of order n over \mathbb{F}_q :

Definition 14. Let $\mathcal{U} \in \mathcal{G}_q(k, n)$ and G be a subgroup of GL_n . Then

$$\mathcal{U}G = \{\mathcal{U}A \mid A \in G\}$$

is called the *orbit code* generated by the initial point \mathcal{U} and the group G .

As shown in [37], orbit codes can be seen as the analog of linear codes in classical block coding. Their structure can be used for an easy computation of the minimum distance of a code, as well as for decoding algorithms (e.g. one can coset-leader decode them). For more information on orbit codes the interested reader is referred to [25], [28], [34], [37].

Example 15. Let $q = k = m = 2$ and $n = 3$. Moreover, let

$$\mathcal{U} = \text{rs} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and G be the cyclic group generated by the matrix

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The corresponding orbit code

$$\mathcal{U}G = \{\mathcal{U}P^i \mid i = 0, \dots, |G| - 1\} = \left\{ \text{rs} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \text{rs} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \right\} \subset \mathcal{G}_2(2, 3)$$

has 2 elements and minimum subspace distance 2.

One can also use the orbit code construction to construct spread codes, as illustrated in the following example.

Example 16. Let $q = k = m = 2$ and $n = 4$. Moreover, let

$$\mathcal{U} = \text{rs} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

and

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

be the companion matrix of the irreducible polynomial $x^4 + x + 1 \in \mathbb{F}_2[x]$. The group $G = \langle P \rangle$ is a subgroup of GL_4 of cardinality 15. The corresponding orbit code $\mathcal{U}G = \{\mathcal{U}P^i \mid i = 0, \dots, 14\} \subset \mathcal{G}_2(2, 4)$ has 5 elements and minimum subspace distance 4. Hence, it is a spread code in $\mathcal{G}_2(2, 4)$.

The following lemma gives a general construction of a spread in $\mathcal{G}_q(k, n)$ as a cyclic orbit code. This construction is again well-known and can be found e.g. in [1], [37].

Lemma 17. Let $P \in \text{GL}_n$ be a companion matrix of a monic primitive polynomial $p(x) \in \mathbb{F}_q[x]$ of degree n . Moreover, let $\mathcal{U} \in \mathcal{G}_q(k, n)$ be the vector space representation of the subfield \mathbb{F}_{q^k} of \mathbb{F}_{q^n} . Then $\mathcal{C} = \mathcal{U}\langle P \rangle$ is a spread code in $\mathcal{G}_q(k, n)$.

C. Message encoding and retrieval

We can define message encoding and retrieval maps very general, for any type of code, as follows.

Definition 18. For a given code \mathcal{C} in some space X and some message space \mathcal{M} , an *encoding map* for the code \mathcal{C}

$$\text{enc} : \mathcal{M} \longrightarrow X$$

is an injective map with image \mathcal{C} . The inverse map

$$\text{enc}^{-1} : \mathcal{C} \longrightarrow \mathcal{M}$$

is called the corresponding *message retrieval map*.

In our setting of subspace codes, $X = \mathcal{P}_q(n)$, or if we use only constant dimension codes, $X = \mathcal{G}_q(k, n)$.

Mostly in the information theory literature, a general message set is represented as $\mathcal{M} = \{0, \dots, |\mathcal{C}| - 1\}$. For classical linear block codes in \mathbb{F}_q^n the usual message space is $\mathcal{M} = \mathbb{F}_q^k$ for some integer $k \leq n$. With the q -adic expansion this can easily be translated to the message set $\{0, \dots, q^k - 1\}$. In contrast, not any set of integers $\{0, \dots, j - 1\}$ can be bijectively mapped to some linear vector space. In particular, if j is not a prime power, there exists no linear vector space over a finite field of the same cardinality. In this paper, since the cardinalities of our codes are in general not prime powers, we derive encoding maps for message sets of the form $\mathcal{M} = \{0, \dots, |\mathcal{C}| - 1\}$.

In the subspace coding case it is not obvious in general, how message encoding or message retrieval can be done. However, an elegant solution is given for the Reed-Solomon-like codes in [21]. For such a code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ the message space is

$$\mathcal{M} = \mathbb{F}_{q^{n-k}}^{k - \frac{d_S(\mathcal{C})}{2} + 1},$$

which is isomorphic (as a vector space) to $\mathbb{F}_q^{(n-k)(k - \frac{d_S(\mathcal{C})}{2} + 1)}$, and an encoding map for \mathcal{C} is given by

$$\begin{aligned} \text{enc} : \mathbb{F}_{q^{n-k}}^{k - \frac{d_S(\mathcal{C})}{2} + 1} &\longrightarrow \mathcal{G}_q(k, n) \\ (u_1, \dots, u_{k - \frac{d_S(\mathcal{C})}{2} + 1}) &\longmapsto \langle (\psi_k^{-1}(\beta_j), \psi_{n-k}^{-1}(\sum_{i=0}^{k - \frac{d_S(\mathcal{C})}{2}} u_{i+1}\beta_j^{q^i})) \mid j = 1, \dots, k \rangle \end{aligned}$$

where $\beta_1, \dots, \beta_k \in \mathbb{F}_{q^{n-k}}$ are linearly independent over \mathbb{F}_q , and we use the isomorphisms $\langle \beta_1, \dots, \beta_k \rangle \cong \mathbb{F}_q^k$ and $\mathbb{F}_{q^{n-k}} \cong \mathbb{F}_q^{n-k}$ for the two vector entries on the right side, respectively. Via interpolation this map is invertible and the inverse is computable in polynomial time. Hence, one gets a feasible message retrieval map as well. In fact, in the decoding algorithm of [21], error correction and message retrieval is done in one algorithm.

As already mentioned in the introduction, many subspace codes cannot be represented as a linear block code over some extension field, hence the idea from above is not necessarily adaptable to other subspace code constructions. Therefore we will use other, different approaches to derive encoding and retrieval maps for the two classes of codes we will investigate in this paper.

One of the ideas we will pursue, is to use *enumerative coding* for message encoding for constant dimension codes. Enumerative coding for the Grassmannian was studied in [30], where the idea of enumerative source encoding of q -ary block codes of Cover [4] was translated to a subspace setting. In contrast to our contribution, the algorithms of [30] are only stated for the whole Grassmannian, and not for any error-correcting constant dimension codes. The idea of enumerative coding was also used in [29] to encode subspace Gray codes. These codes are, however, no-error-correcting. To adapt the ideas of enumerative coding to spread or orbit codes, one needs an efficiently computable map that counts the number of subspaces whose reduced row echelon form fulfills certain requirements. This is feasible for Desarguesian spread codes and will be explained in Subsection IV-B. For orbit codes however, it is not clear how such a map could efficiently be computed, which is why we will not pursue the idea of enumerative coding as an encoding map in this context.

III. COMPUTATIONAL PRELIMINARY RESULTS

In this section we derive complexity orders of tasks we will need in our main algorithms in Sections IV and V. For comparability with error decoding complexities we will do our complexity analyses over \mathbb{F}_q , which is why we will represent the messages $0, 1, \dots, |C| - 1$ in their q -adic expansion. For simplicity we represent these q -adic expansions in \mathbb{F}_q^n , although not all coordinates are necessarily needed.

We use the Big-O notation for the computational complexities of our algorithms, where we use the index q to specify that the given complexity order is over the base field \mathbb{F}_q and not over some extension field.

The following results are well-known.

Lemma 19. *Consider \mathbb{F}_q and an extension field \mathbb{F}_{q^k} . Represent $\mathbb{F}_{q^k} \cong \mathbb{F}_q[\alpha]$ for some suitable α .*

- 1) *Multiplying two elements from \mathbb{F}_{q^k} can be done with $\mathcal{O}_q(k^2)$ operations in \mathbb{F}_q . The same holds for division in \mathbb{F}_{q^k} .*
- 2) *Let $\beta \in \mathbb{F}_{q^k}$ and $0 \leq i \leq q^k - 1$. Computing the modular exponentiation β^i of $\beta \in \mathbb{F}_{q^k}$, i.e., finding the representation of β^i in the basis $\{1, \alpha, \dots, \alpha^{k-1}\}$ of \mathbb{F}_{q^k} , can be done with at most $\mathcal{O}_q(k^3)$ operations in \mathbb{F}_q .*

Proof:

- 1) Any element in \mathbb{F}_{q^k} can be represented as a polynomial over \mathbb{F}_q of degree less than k . Since polynomial multiplication and division of polynomials of degree at most k can be done with $\mathcal{O}_q(k^2)$ operations (see e.g. [14, Corollary 4.6]), the first statement follows.
- 2) Using a normal basis of \mathbb{F}_{q^k} , it was shown in [13] that modular exponentiation can be done with $k/(\log_q k)$ multiplications in \mathbb{F}_{q^k} . The change of basis to the normal basis is a linear map and can hence be done with $\mathcal{O}_q(k^2)$ operations in \mathbb{F}_q . Together with 1) the statement follows. ■

Lemma 20. *Let α be a root of a monic irreducible polynomial $p(x) \in \mathbb{F}_q[x]$ of degree k , such that $\mathbb{F}_{q^k} \cong \mathbb{F}_q[\alpha]$. The complexity of computing the map $\psi_k : \mathbb{F}_{q^k}^k \rightarrow \mathbb{F}_q[\alpha]$, as well as computing its inverse, is in $\mathcal{O}_q(k)$.*

Proof: Let $u = (u_1, u_2, \dots, u_k) \in \mathbb{F}_{q^k}^k$. If we want to compute $\psi(u)$ we simply need to write the k vector coordinates u_1, \dots, u_k as polynomial coefficients in $\sum_{i=1}^k u_i \alpha^{i-1}$. The inverse map can be computed analogously, by writing the polynomial coefficients as vector entries. Therefore both maps can be computed with a complexity in $\mathcal{O}_q(k)$. ■

Lemma 21. *The complexity of computing the map $\phi_{k,m} : \mathbb{F}_{q^k}^m \rightarrow \{0, 1, \dots, q^{km} - 1\}$, as well as computing its inverse map $\phi_{k,m}^{-1}$, is in $\mathcal{O}_q(km) = \mathcal{O}_q(n)$.*

Proof: Recall that $\phi_{k,m}(u_1, u_2, \dots, u_m) = \sum_{i=1}^m \varphi(u_i) q^{k(i-1)}$ and that we represent the integers in their q -adic expansion in \mathbb{F}_q^n . Denote by (i_1, i_2, \dots, i_n) the q -adic expansion of the integer i . Then $\phi_{k,m}^{-1}(i_1, i_2, \dots, i_n) = (\psi_k(i_1, i_2, \dots, i_k), \dots, \psi_k(i_{n-k+1}, i_{n-k+2}, \dots, i_n))$. It follows from Lemma 20 that the complexity of computing $\phi_{k,m}^{-1}$ is in $\mathcal{O}_q(mk) = \mathcal{O}_q(n)$.

Similarly, we get $\phi_{k,m}(u_1, u_2, \dots, u_m) = (\psi_k^{-1}(u_1), \psi_k^{-1}(u_2), \dots, \psi_k^{-1}(u_m))$, which is the respective integer in its q -adic expansion. By Lemma 20 this can again be done with a complexity in $\mathcal{O}_q(n)$. ■

The next task we will investigate is computing powers of companion matrices of irreducible polynomials.

Lemma 22. *Let $p(x) \in \mathbb{F}_q[x]$ be monic irreducible of degree k , α a root of $p(x)$, and let $P \in \text{GL}_k$ be its companion matrix. Then P^i can be computed with a complexity of at most $\mathcal{O}_q(k^3)$ operations over \mathbb{F}_q .*

If moreover the representation of α^i in the basis $\{1, \alpha, \dots, \alpha^{k-1}\}$ of $\mathbb{F}_q[\alpha]$ is known, then P^i can be computed with a complexity in $\mathcal{O}_q(k^2)$.

Proof: Recall from Section II that $\psi_k(uP) = \psi_k(u)\alpha$ for any $u \in \mathbb{F}_q^k$. Thus, if we apply ψ_k on the rows of P^i , we get

$$\psi_k(P^i) = \psi_k(P)\alpha^{i-1} = \begin{pmatrix} \alpha \\ \alpha^2 \\ \alpha^3 \\ \vdots \\ \alpha^k \end{pmatrix} \alpha^{i-1} = \begin{pmatrix} \alpha^i \\ \alpha^{i+1} \\ \alpha^{i+2} \\ \vdots \\ \alpha^{i+k-1} \end{pmatrix}.$$

For $0 \leq i \leq k-1$ we have $\psi_k^{-1}(\alpha^i) = e_{i+1}$, where $e_i \in \mathbb{F}_q^k$ is the i -th unit vector. For higher values of i we need to compute the representation of α^i in the basis $\{1, \alpha, \dots, \alpha^{k-1}\}$. This representation, if not known, can be computed with a complexity of at most $\mathcal{O}_q(k^3)$ (see Lemma 19).

Then we can construct P^i as follows: The first row is simply the vector representation $\psi_k^{-1}(\alpha^i) \in \mathbb{F}_q^k$ of $\alpha^i \in \mathbb{F}_q[\alpha]$. This can be done with k coefficient transfers (over \mathbb{F}_q). For $2 \leq j \leq k$ the j -th row of P^i , denoted by P_j^i , is given by

$$P_j^i = P_{j-1}^i P = (0, (P_{j-1}^i)_{[1, k-1]}) + (P_{j-1}^i)_k \cdot (-p_0, -p_1, \dots, -p_{k-1}),$$

where $(P_{j-1}^i)_{[1, k-1]}$ denotes the subvector of P_{j-1}^i without the last coordinate and $(P_{j-1}^i)_k$ denotes the last coordinate of P_{j-1}^i . Hence, for each row of P^i we need to multiply a vector in \mathbb{F}_q^k by a scalar from \mathbb{F}_q and then add two vectors from \mathbb{F}_q^k . Both of these computations need k operations in \mathbb{F}_q . Since we need to do this for each row of P^i , we get an overall complexity order of $\mathcal{O}_q(k^2)$, if the representation of α^i in the basis $\{1, \alpha, \dots, \alpha^{k-1}\}$ is known. If this representation of α^i is not known, the overall complexity becomes $\mathcal{O}_q(k^3)$. ■

We can now derive the complexity of computing the map des , for constructing a Desarguesian spread code, as defined in Construction I in Section II:

Theorem 23. *Consider the map $\text{des} : \mathcal{G}_{q^k}(1, m) \rightarrow \mathcal{G}_q(k, n)$, whose image is a Desarguesian spread $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$. The map des and its inverse $\text{des}^{-1} : \mathcal{C} \rightarrow \mathcal{G}_{q^k}(1, m)$ can be computed with a complexity order in $\mathcal{O}_q(kn)$.*

Proof: As before, let $p(x) \in \mathbb{F}_q[x]$ be monic irreducible of degree k , α be a root of $p(x)$, P the corresponding companion matrix and let ρ be the isomorphism from $\mathbb{F}_q[\alpha]$ to $\mathbb{F}_q[P]$. For the computation of the map des , take the normalized representation of the preimage $(u_1, \dots, u_m) \in \mathcal{G}_{q^k}(1, m)$ and consider the elements $u_i \in \mathbb{F}_{q^k} \cong \mathbb{F}_q[\alpha]$. For each $i \in \{1, \dots, m\}$ such that $u_i \neq 0$, use the construction used in the proof of Lemma 22 to construct $P_i = \rho(u_i)$ for $i = 1, 2, \dots, m$. By Lemma 22, this can be done with a complexity in $\mathcal{O}_q(k^2)$. Then

$$\text{rs}(\rho(u_1), \rho(u_2), \dots, \rho(u_m)) = \text{rs}(P_1, P_2, \dots, P_m) \in \mathcal{G}_q(k, n)$$

is the respective spread codeword. Since we need to construct at most $m = n/k$ matrices P_i , we get an overall complexity order of $\mathcal{O}_q(mk^2) = \mathcal{O}_q(kn)$.

We now consider the inverse map des^{-1} . Choose one vector $v \in \mathbb{F}_q^n$ of the given codeword $\mathcal{U} \in \mathcal{G}_q(k, n)$ and represent it as $u \in \mathbb{F}_{q^k}^m$ via

$$u = (\psi_k(v_1, v_2, \dots, v_k), \psi_k(v_{k+1}, v_{k+2}, \dots, v_{2k}), \dots, \psi_k(v_{n-k+1}, v_{n-k+2}, \dots, v_n)).$$

By Lemma 20 this representation can be done with $\mathcal{O}_q(mk) = \mathcal{O}_q(n)$ operations. Normalize u by dividing all coordinates by the first non-zero entry of u . This normalized vector is the representative of the respective element in $\mathcal{G}_{q^k}(1, m)$. For the normalization, one needs at most m divisions over \mathbb{F}_{q^k} . Each such division can be done with $\mathcal{O}_q(k^2)$ operations (see Lemma 19), i.e., we get an overall complexity of $\mathcal{O}_q(mk^2) = \mathcal{O}_q(kn)$. ■

IV. MESSAGE ENCODING FOR DESARGUESIAN SPREAD CODES

In this section we derive message encoding and retrieval maps for Desarguesian spread codes. In the first subsection we derive an intuitive encoder for these type of codes, arising from the Grassmannian representation in Construction I. In the second subsection we use the idea of enumerative coding on the Grassmannian to derive an encoder for Desarguesian spread codes. We then show that this second encoder is the same as the encoder from the first subsection with a little twist.

A. Ad Hoc Construction

We will now derive a message encoding map by concatenating the map des with an injective map f from $\{0, \dots, (q^n - 1)/(q^k - 1) - 1\}$ to $\mathcal{G}_{q^k}(1, m)$. This map is defined as follows:

$$f : \{0, \dots, (q^n - 1)/(q^k - 1) - 1\} \longrightarrow \mathcal{G}_{q^k}(1, m)$$

$$i \longmapsto \text{rs}\left(\underbrace{0, \dots, 0}_{m - \epsilon(i) - 1}, 1, \phi_{k, \epsilon(i)}^{-1}\left(i - \sum_{j=0}^{\epsilon(i)-1} q^{jk}\right)\right).$$

where $\epsilon(i) := \min\{\ell \mid \sum_{j=0}^{\ell} q^{jk} \geq i + 1\}$ and $\phi_{k,\epsilon(i)} : \mathbb{F}_{q^k}^{\epsilon(i)} \rightarrow \{0, \dots, q^{k\epsilon(i)} - 1\}$ is the inverse q^k -adic expansion, as explained in Section II. We defined $\epsilon(i)$ such that f behaves as follows:

$$\begin{aligned} 0 &\mapsto \text{rs}(0, \dots, 0, 0, 0, 1), \\ \{1, 2, \dots, q^k\} &\ni i \mapsto \{\text{rs}(0, \dots, 0, 0, 1, v) \mid v \in \mathbb{F}_{q^k}, v = \phi_{k,1}^{-1}(i - 1)\}, \\ \{q^k + 1, q^k + 2, \dots, q^{2k} + q^k\} &\ni i \mapsto \{\text{rs}(0, \dots, 0, 1, v) \mid v \in \mathbb{F}_{q^k}, v = \phi_{k,2}^{-1}(i - q^k - 1)\}, \\ &\vdots \\ \left\{ \sum_{j=1}^{m-2} q^{jk} + 1, \sum_{j=1}^{m-2} q^{jk} + 2, \dots, \sum_{j=1}^{m-1} q^{jk} \right\} &\ni i \mapsto \left\{ \text{rs}(1, v) \mid v \in \mathbb{F}_{q^k}^{m-1}, v = \phi_{k,m-1}^{-1}\left(i - \sum_{j=0}^{m-2} q^{jk}\right) \right\}. \end{aligned}$$

Theorem 24. *The map f is bijective and hence*

$$\text{enc}_1 := \text{des} \circ f$$

is an injective map from $\{0, \dots, (q^n - 1)/(q^k - 1) - 1\}$ to $\mathcal{G}_q(k, n)$, whose image is the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ from Construction I. Therefore, enc_1 is an encoding map for the respective Desarguesian spread code \mathcal{C} .

Proof: By the above shown behavior of f and the fact that $\phi_{k,\epsilon(i)}$ is bijective, it follows that f is injective. Since $(q^n - 1)/(q^k - 1) = \sum_{j=0}^{m-1} q^{jk}$, the cardinalities of domain and codomain of f are equal. This implies that f is bijective. Since the image of des is the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$, the statement follows. ■

We will now give two algorithms, describing how to compute the encoding map enc_1 and the respective message retrieval map enc_1^{-1} . As before we denote by ρ the isomorphisms from $\mathbb{F}_q[\alpha]$ to $\mathbb{F}_q[P]$. The computational complexity order of these two algorithms is afterwards given in Theorem 31.

Algorithm 1 Message encoding for the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ from Construction I.

Require: A message $i \in \{0, 1, \dots, (q^n - 1)/(q^k - 1) - 1\}$.

Compute $\epsilon(i)$.

Compute $i' = i - \sum_{j=0}^{\epsilon(i)-1} q^{jk}$.

Compute $u' = \phi_{k,\epsilon(i)}^{-1}(i')$.

Set $u := (\underbrace{0, \dots, 0}_{m-\epsilon(i)-1}, 1, u')$.

Set $U := (\rho(u_1), \rho(u_2), \dots, \rho(u_m))$.

return $\mathcal{U} = \text{rs}(U)$

Algorithm 2 Message retrieval for the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ from Construction I.

Require: A spread codeword $\mathcal{U} \in \mathcal{C}$.

Choose a non-zero vector $v \in \mathcal{U}$.

Compute $u = (\psi_k(v_1, v_2, \dots, v_k), \psi_k(v_{k+1}, v_{k+2}, \dots, v_{2k}), \dots, \psi_k(v_{n-k+1}, v_{n-k+2}, \dots, v_n))$.

Normalize u .

Set $\epsilon(i) := m -$ (the coordinate of the first non-zero entry of u).

Set $u' :=$ the rightmost $\epsilon(i)$ coordinates of u .

Compute $i = \phi_{k,\epsilon(i)}(u') + \sum_{j=0}^{\epsilon(i)-1} q^{jk}$.

return i

Theorem 25. *Algorithms 1 and 2 compute the images of the maps enc_1 and enc_1^{-1} , respectively, for any valid input.*

Proof: Algorithm 1 first computes the image $f(i)$ in the first three steps and then computes the image of this result under the map des . Hence, by Theorem 24, Algorithm 1 returns the corresponding Desarguesian spread codeword to the input.

In Algorithm 2 we note that the choice of the non-zero vector $v \in \mathcal{C}$ does not matter, since any choice will result in the same normalized $u \in \mathbb{F}_{q^k}^m$. One can easily check that the last three steps of the algorithm then compute $f^{-1}(u)$. For this note that in the definition of f one can see that $\epsilon(i)$ is equal to m minus the coordinate of the first non-zero entry of u . Hence, by Theorem 24, Algorithm 2 returns the message corresponding to a codeword of the Desarguesian spread \mathcal{C} . ■

Example 26. Let $q = k = 2$ and $m = 3$. Moreover, let $\alpha \in \mathbb{F}_{2^2}$ be a primitive element and let the elements of \mathbb{F}_{2^2} be identified via the map φ with $0 \mapsto 0, 1 \mapsto 1, \alpha \mapsto 2, \alpha^2 = \alpha + 1 \mapsto 3$. Moreover, let P be the companion matrix of α (see

Example 13). We want to encode the message $i = 14$. Following Algorithm 1, we compute $\epsilon(i) = 2$, $i' = 14 - (1 + 4) = 9$ and $u' = \phi_{2,2}^{-1}(9) = (1, \alpha)$. Thus we get $u = (1, 1, \alpha)$ and $U = (I_2, I_2, P)$ as the basis matrix of the respective codeword.

Similarly, we get that the first 10 non-negative integers are mapped by f to the following elements of $\mathcal{G}_2(1, 3)$:

i	0	1	2	3	4	5	6	7	8	9	...
$f(i)$	rs(0, 0, 1)	rs(0, 1, 0)	rs(0, 1, 1)	rs(0, 1, α)	rs(0, 1, α^2)	rs(1, 0, 0)	rs(1, 1, 0)	rs(1, α , 0)	rs(1, α^2 , 0)	rs(1, 0, 1)	...

Following Algorithm 1, the matrix representation of the spread code elements in $\mathcal{G}_2(2, 6)$ are given by replacing the elements of $\mathbb{F}_2[\alpha]$ by the respective element of $\mathbb{F}_2[P]$.

Example 27. Consider the same setting as in Example 26. Let $\mathcal{U} = \text{rs}(\rho(1), \rho(\alpha + 1), \rho(1)) \in \mathcal{C}$ be a codeword, for which we would like to find the corresponding message. Following Algorithm 2, we choose some non-zero $v \in \mathcal{U}$, say $v = (0, 1, 1, 1, 0, 1)$, and compute $u = (\alpha, 1, \alpha)$. Then we normalize u to $(1, \alpha + 1, 1)$. Since the first non-zero entry is in position 1, we get $\epsilon(i) = 3 - 1 = 2$. Then we compute $i = \phi_{2,2}(\alpha + 1, 1) + \sum_{j=0}^1 2^{2j} = (3 \cdot 1 + 1 \cdot 4) + (1 + 4) = 12$.

In the following we analyze the complexity of computing enc_1 and enc_1^{-1} , i.e., Algorithms 1 and 2. For this we need the following lemma, which implies that the computation of $\epsilon(i)$ can be done efficiently in the q -adic expansion.

Lemma 28. For any $\ell < m$, the q -adic expansion $(u_1, \dots, u_n) \in \mathbb{F}_q^n$ of the integer $\sum_{j=0}^{\ell} q^{jk}$ is given by

$$\phi_{1,n}^{-1}\left(\sum_{j=0}^{\ell} q^{jk}\right) = (u_1, \dots, u_n), \quad \text{where } \begin{cases} u_i = 1 & \text{if } k|(i-1) \text{ and } i-1 \leq \ell k \\ u_i = 0 & \text{else} \end{cases}.$$

Proof: The inverse q -adic expansion maps (u_1, \dots, u_n) to $\sum_{i=1}^n \varphi(u_i)q^{i-1}$. With the above values of u_1, \dots, u_n one gets (recall that $\varphi(0) = 0$ and $\varphi(1) = 1$)

$$\phi_{1,n}(u_1, u_2, \dots, u_n) = \sum_{i=1}^n \varphi(u_i)q^{i-1} = q^0 + q^k + q^{2k} + \dots + q^{\ell k} = \sum_{j=0}^{\ell} q^{jk}.$$

■

To check if an integer i is greater than an integer j , one needs to check if the q -adic expansion of i is greater than the q -adic expansion of j in reverse lexicographic order.

Example 29. Consider the setting of Example 26. We represent the message set $\{0, 1, \dots, 20\}$ in their 2-adic expansion:

$$\begin{array}{llll} 0 \rightarrow (0, 0, 0, 0, 0, 0) & 2 \rightarrow (0, 1, 0, 0, 0, 0) & \dots & 19 \rightarrow (1, 1, 0, 0, 1, 0) \\ 1 \rightarrow (1, 0, 0, 0, 0, 0) & 3 \rightarrow (1, 1, 0, 0, 0, 0) & & 20 \rightarrow (0, 0, 1, 0, 1, 0). \end{array}$$

By Lemma 28 the 2-adic expansion of $\sum_{j=0}^{\ell} 2^{2j}$ for $\ell \in \{0, 1, 2\}$ are given by:

$$\begin{aligned} \ell = 0 : & (1, 0, 0, 0, 0, 0) \\ \ell = 1 : & (1, 0, 1, 0, 0, 0) \\ \ell = 2 : & (1, 0, 1, 0, 1, 0). \end{aligned}$$

Hence, we can compute $\epsilon(i) = \min\{\ell \mid \sum_{j=0}^{\ell} 2^{2j} \geq i+1\} = \min\{\ell \mid \sum_{j=0}^{\ell} 2^{2j} - 1 \geq i\}$ for a given message $i \in \{0, 1, \dots, 20\}$ in its 2-adic expansion $u = (u_1, \dots, u_6) \in \mathbb{F}_2^6$ as follows (note that $u_6 = 0$ for all messages):

- If $u = 0$, then $\epsilon(i) = 0$.
- If $u \neq 0$, check coordinate-wise (from right to left) if u is less than or equal to $(0, 0, 1, 0, 0, 0)$. If so, then $\epsilon(i) = 1$, otherwise $\epsilon(i) = 2$.

In the proof of the following lemma we describe how to compute $\epsilon(i)$ in general.

Lemma 30. The complexity of computing $\epsilon(i)$, for $i \in \{0, 1, \dots, q^n - 1\}$, is in $\mathcal{O}_q(n)$.

Proof: If $i = 0$, then $\epsilon(i) = 0$. Assume now that $i > 0$. Represent i in its q -adic expansion $\phi_{1,n}^{-1}(i) = (u_1, \dots, u_n) = u \in \mathbb{F}_q^n$. Find the first non-zero coordinate from the right j^* of u , i.e., $j^* = \min\{i \mid u_{n-i} = 0\}$.

- If $k \nmid (j^* - 1)$, then $\epsilon(i) = \lceil (j^* - 1)/k \rceil$.
- If $k \mid (j^* - 1)$, then compare u coordinate-wise (from right to left) with the q -adic expansion of $\sum_{j=0}^{(j^*-1)/k} q^{jk} - 1$, as illustrated in Lemma 28. If u is strictly greater than the q -adic expansion of $\sum_{j=0}^{(j^*-1)/k} q^{jk} - 1$, then $\epsilon(i) = (j^* - 1)/k + 1$, otherwise $\epsilon(i) = (j^* - 1)/k$.

Thus we need at most n coordinate comparisons, and a division over \mathbb{F}_q , which implies the statement. ■

We can now give the computational complexity order of the message encoding and retrieval for Desarguesian spread codes:

Theorem 31. Algorithms 1 and 2 have a computational complexity in $\mathcal{O}_q(kn)$.

Proof: For the encoder, i.e., Algorithm 1, the complexity of computing $f(i)$ is dominated by finding $\epsilon(i)$ and computing $\phi_{k,\epsilon(i)}^{-1}(i - \sum_{j=0}^{\epsilon(i)-1} q^{jk})$. By Lemmas 20 and 30 these task can be done with $\mathcal{O}_q(n)$ operations. Since we know from Theorem 23 that des can be computed with a computational complexity of order $\mathcal{O}_q(kn)$, the statement for enc_1 follows.

Algorithm 2 describes how to compute the retrieval map enc_1^{-1} . By Lemma 20, getting $u \in \mathbb{F}_{q^k}^m$ from $v \in \mathbb{F}_q^n$ needs $\mathcal{O}_q(mk) = \mathcal{O}_q(n)$ operations. Analogously to the normalization in the proof of Theorem 23, the normalization of u requires $\mathcal{O}_q(k^2m) = \mathcal{O}_q(kn)$ operations. Then the complexity of the computation of $\epsilon(i)$ become negligible, as does the computation of $\phi_{k,\epsilon(i)}(u')$, by Lemma 21. ■

B. The Enumerative Coding Point of View

In this subsection we want to use the idea of enumerative coding to derive a message encoding and a retrieval map for Desarguesian spread codes.

The method of enumerative coding for block codes $C \subseteq \mathbb{F}_q^n$, as presented by Cover in [4], is as follows: Denote by $\text{enu}_C(u_1, \dots, u_j)$ the number of elements of C , for which the first j coordinates are given by (u_1, \dots, u_j) . Moreover, fix an order $<$ on \mathbb{F}_q . Then the lexicographic index of $u = (u_1, \dots, u_n) \in \mathbb{F}_q^n$ is given by $\text{ind}_C(u) = \sum_{j=1}^n \sum_{y < u_j} \text{enu}_C(u_1, \dots, u_{j-1}, y)$. This indexing function is then a message retrieval map.

This method of enumerative coding was adapted to the whole Grassmannian space in [30]. In their setting, the enumerating function counts all vector spaces in $\mathcal{G}_q(k, n)$, whose reduced row echelon forms fulfill certain conditions.

For our purpose of deriving a message encoder for Desarguesian spread codes in $\mathcal{G}_q(k, n)$, we use the idea of [4] on $\mathcal{G}_{q^k}(1, m)$. I.e., we define the enumerating function $\text{enu}_m(u_1, \dots, u_j)$ to count all elements of $\mathcal{G}_{q^k}(1, m)$, whose normalized representations have (u_1, \dots, u_j) as their first j entries. Furthermore we need to fix a bijection between \mathbb{F}_{q^k} and the set of integers $\{0, \dots, q^k - 1\}$, where we choose the map φ , as defined in Section II, as this bijection. This map induces an order $<$ on \mathbb{F}_{q^k} . The indexing function for enumerative coding on $\mathcal{G}_{q^k}(1, m)$ is now given by

$$\begin{aligned} \text{ind} : \mathcal{G}_{q^k}(1, m) &\longrightarrow \{0, \dots, (q^n - 1)/(q^k - 1) - 1\} \\ \text{rs}(u_1, \dots, u_m) &\longmapsto \sum_{j=1}^m \sum_{y < u_j} \text{enu}_m(u_1, \dots, u_{j-1}, y) \end{aligned}$$

where (u_1, \dots, u_m) is the normalized basis vector of the preimage. This function can easily be extended to a message retrieval map for Desarguesian spread codes, as shown in the following.

Theorem 32. *The map $\text{des} \circ \text{ind}^{-1}$ is an encoding map for the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ from Construction I. Its inverse $\text{ind} \circ \text{des}^{-1}$ is the corresponding message retrieval map.*

Proof: Since the Desarguesian spread code in $\mathcal{G}_q(k, n)$ from Construction I is isomorphic to $\mathcal{G}_{q^k}(1, m)$, we simply need to show that ind is a bijection. This can be done in analogy to Cover's original proof in [4]: If $y_1 < y_2$, then $\text{ind}(u_1, \dots, u_{j-1}, y_1, *, \dots, *) < \text{ind}(u_1, \dots, u_{j-1}, y_2, *, \dots, *)$ for any $y_1, y_2 \in \mathbb{F}_{q^k}$. Thus ind is injective. We now show that the image is $\{0, \dots, (q^n - 1)/(q^k - 1) - 1\}$. The element in the preimage of lowest index is $\text{ind}(0, 0, \dots, 0, 1)$, which has index 0. Let y_{\max} be the largest element of \mathbb{F}_{q^k} with respect to $<$. Then the preimage with the highest index is $(1, y_{\max}, \dots, y_{\max})$, for which ind counts all other elements of $\mathcal{G}_{q^k}(1, m)$. Hence ind takes on every value in $\{0, \dots, (q^n - 1)/(q^k - 1) - 1\}$. I.e., ind is bijective, and since $\text{des} : \mathcal{G}_{q^k}(1, m) \rightarrow \mathcal{G}_q(k, n)$ is an injective map, whose image is the spread code, the statement follows. ■

It remains to investigate how the map enu_m can be efficiently computed.

Lemma 33. *Let $j \leq m$ and $(u_1, \dots, u_j) \in \mathbb{F}_{q^k}^j$ such that the first non-zero entry, if existent, is equal to 1. Then*

$$\text{enu}_m(u_1, \dots, u_j) = \begin{cases} \frac{q^{k(m-j)} - 1}{q^k - 1} & \text{if } u_1 = \dots = u_j = 0 \\ q^{k(m-j)} & \text{else} \end{cases}.$$

Proof: If (u_1, \dots, u_j) is all-zero, then the remaining $m - j$ entries of any completion in $\mathcal{G}_{q^k}(1, m)$ need to be elements of $\mathcal{G}_{q^k}(1, m - j)$. The number of such elements is exactly $(q^{k(m-j)} - 1)/(q^k - 1)$. On the other hand, if (u_1, \dots, u_j) is not all-zero, then any completion is already normalized (since we assume that (u_1, \dots, u_j) itself is normalized) and the remaining entries can be any element of \mathbb{F}_{q^k} . This implies the formula. ■

We can now simplify the indexing function as follows.

Corollary 34. *Let $\text{rs}(u_1, \dots, u_m) \in \mathcal{G}_{q^k}(1, m)$. Define $j_1 := \min\{j \mid u_j \neq 0\}$ as the coordinate of the first non-zero entry of*

(u_1, \dots, u_m) . Then

$$\begin{aligned} \text{ind}(\text{rs}(u_1, \dots, u_m)) &= \frac{q^{k(m-j_1)} - 1}{q^k - 1} + \sum_{j=j_1+1}^m \varphi(u_j) q^{k(m-j)} \\ &= \sum_{j=0}^{m-j_1-1} (\varphi(u_{m-j}) + 1) q^{kj}. \end{aligned}$$

Proof: Since we assume that all vectors are normalized, we have $u_{j_1} = 1$ and $u_1 = u_2 = \dots = u_{j_1-1} = 0$. We can use Lemma 33 and rewrite

$$\begin{aligned} \sum_{j=1}^m \sum_{y < u_j} \text{enu}_m(u_1, \dots, u_{j-1}, y) &= \sum_{j=1}^{j_1} \sum_{y < u_j} \text{enu}_m(u_1, \dots, u_{j-1}, y) + \sum_{j=j_1+1}^m \sum_{y < u_j} \text{enu}_m(u_1, \dots, u_{j-1}, y) \\ &= \text{enu}_m(\underbrace{0, 0, \dots, 0}_{j_1}, 0) + \sum_{j=j_1+1}^m \sum_{y < u_j} \text{enu}_m(u_1, \dots, u_{j-1}, y) \\ &= \frac{q^{k(m-j_1)} - 1}{q^k - 1} + \sum_{j=j_1+1}^m \varphi(u_j) q^{k(m-j)} \\ &= \sum_{j=0}^{m-j_1-1} q^{kj} + \sum_{j=0}^{m-j_1-1} \varphi(u_{m-j}) q^{kj} \\ &= \sum_{j=0}^{m-j_1-1} (\varphi(u_{m-j}) + 1) q^{kj}. \end{aligned}$$

■

Example 35. Let $q = k = 2$ and $m = 3$. Moreover, let $\alpha \in \mathbb{F}_{2^2}$ be a primitive element and let the elements of \mathbb{F}_{2^2} be identified via the map φ with $0 \mapsto 0, 1 \mapsto 1, \alpha \mapsto 2, \alpha^2 \mapsto 3$. Then

$$\begin{aligned} \text{ind}(\text{rs}(0, 0, 1)) &= 0 + 0 = 0 & \text{ind}(\text{rs}(1, 0, 1)) &= 5 + 1 = 6 \\ \text{ind}(\text{rs}(0, 1, \alpha)) &= 1 + 2 = 3 & \text{ind}(\text{rs}(1, \alpha^2, 1)) &= 5 + 12 + 1 = 18. \end{aligned}$$

Note that the two results on the right differ from those seen in Example 26.

The following theorem and corollary show that the enumerative encoder is equal to the previously described encoder enc_1 , if we replace $\phi_{k, \epsilon(i)}$ with $\bar{\phi}_{k, \epsilon(i)}$ in the definition of f , where

$$\begin{aligned} \bar{\phi}_{k, \epsilon(i)} : \mathbb{F}_{q^k}^{\epsilon(i)} &\longrightarrow \{0, 1, \dots, q^{k\epsilon(i)-1}\} \\ (u_1, u_2, \dots, u_{\epsilon(i)}) &\longmapsto \sum_{j=0}^{\epsilon(i)-1} \varphi(u_{m-j}) q^{kj} \end{aligned}$$

is another bijection from $\mathbb{F}_{q^k}^{\epsilon(i)}$ to $\{0, 1, \dots, q^{k\epsilon(i)-1}\}$. We call the new map, i.e., f after replacing $\phi_{k, \epsilon(i)}$ with $\bar{\phi}_{k, \epsilon(i)}$, \bar{f} .

Theorem 36. The map $\bar{f}^{-1} : \mathcal{G}_{q^k}(1, m) \rightarrow \{0, 1, \dots, \frac{q^n-1}{q^k-1}\}$ is equal to the map $\text{ind} : \mathcal{G}_{q^k}(1, m) \rightarrow \{0, 1, \dots, \frac{q^n-1}{q^k-1}\}$.

Proof: Let $\text{rs}(u_1, \dots, u_m) \in \mathcal{G}_{q^k}(1, m)$ and let $(u_1, \dots, u_m) \in \mathbb{F}_{q^k}^m$ be its normalized representation. Furthermore let $\epsilon(i)$ be defined as in the definition of f , and let j_1 be defined as in Corollary 34. Then $\epsilon(i) = m - j_1$ and

$$\frac{q^{k(m-j_1)} - 1}{q^k - 1} = \frac{q^{k\epsilon(i)} - 1}{q^k - 1} = \sum_{j=0}^{\epsilon(i)-1} q^{kj}.$$

One knows that $\varphi(u_{m-\epsilon(i)}) = \varphi(u_{j_1}) = 1$ and $\varphi(u_{m-j}) = 0$ for $j > \epsilon(i)$. Hence, together with Corollary 34, we get

$$\begin{aligned} \text{ind}(\text{rs}(u_1, \dots, u_m)) &= \sum_{j=0}^{m-j_1-1} (\varphi(u_{m-j}) + 1) q^{kj} = \sum_{j=0}^{\epsilon(i)-1} q^{kj} + \sum_{j=0}^{\epsilon(i)-1} \varphi(u_{m-j}) q^{kj} \\ &= \sum_{j=0}^{\epsilon(i)-1} q^{kj} + \bar{\phi}_{k, \epsilon(i)}(u_{m-\epsilon(i)+1}, \dots, u_m) = \bar{f}^{-1}(\text{rs}(u_1, \dots, u_m)). \end{aligned}$$

The previous theorem straightforwardly implies the following corollary, that the two encoders (and hence also the respective message retrieval maps) for Desarguesian spread codes from Subsections IV-A and IV-B are equal, if we replace f with \bar{f} in the definition of enc_1 . We denote this second encoder by $\overline{\text{enc}}_1 := \text{des} \circ \bar{f}$.

Corollary 37. *Let $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ be the Desarguesian spread code from Construction I.*

- *The encoding map $\overline{\text{enc}}_1 : \{0, 1, \dots, \frac{q^n-1}{q^k-1}\} \rightarrow \mathcal{G}_q(k, n)$ is equal to the encoding map $\text{des} \circ \text{ind}^{-1} : \{0, 1, \dots, \frac{q^n-1}{q^k-1}\} \rightarrow \mathcal{G}_q(k, n)$.*
- *The message retrieval map $\overline{\text{enc}}_1^{-1} : \mathcal{C} \rightarrow \{0, 1, \dots, \frac{q^n-1}{q^k-1}\}$ is equal to the message retrieval map $\text{ind} \circ \text{des}^{-1} : \mathcal{C} \rightarrow \{0, 1, \dots, \frac{q^n-1}{q^k-1}\}$.*

Even though the two encoders $\overline{\text{enc}}_1$ and $\text{des} \circ \text{ind}^{-1}$ are equal as a map, they give rise to different ways of computing the encoding and the corresponding message retrieval map. Algorithm 3 describes an alternative way of computing the message retrieval for the Desarguesian spread code from Construction I, based on the idea of enumerative coding. The correctness of it follows from Theorem 32 and Corollary 34. We will not give an alternative algorithm for the encoder, since computing the inverse of the indexing function, ind^{-1} , does not give rise to an easier algorithm than Algorithm 1.

Algorithm 3 Message retrieval based on enumerative coding for the Desarguesian spread code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ from Construction I.

Require: A spread codeword $\mathcal{U} \in \mathcal{C}$.

Choose a non-zero vector $v \in \mathcal{U}$.

Compute $u = (\psi_k(v_1, v_2, \dots, v_k), \psi_k(v_{k+1}, v_{k+2}, \dots, v_{2k}), \dots, \psi_k(v_{n-k+1}, v_{n-k+2}, \dots, v_n))$.

Normalize u .

Set $j_1 := \min\{j \mid u_j \neq 0\}$.

Compute $i = \sum_{j=0}^{m-j_1-1} (\varphi(u_{m-j}) + 1) q^{kj}$.

return i

Theorem 38. *The computational complexity of Algorithm 3 is in $\mathcal{O}_q(kn)$.*

Proof: By Lemma 20, getting $u \in \mathbb{F}_{q^k}^m$ from $v \in \mathbb{F}_q^n$ needs $\mathcal{O}_q(mk) = \mathcal{O}_q(n)$ operations. Analogously to the normalization in the proof of Theorem 23, the normalization of u requires $\mathcal{O}_q(k^2m) = \mathcal{O}_q(kn)$ operations. Since we represent i in its q -adic expansion, we have $\varphi(u_i) = \psi_k^{-1}(u_i)$ for $i = 1, 2, \dots, m$. From Corollary 34 we know that the q -adic expansion of i is given by

$$\underbrace{(10 \dots 0 10 \dots 0 \dots 10 \dots 0 00 \dots 0)}_{k} + (\psi_k^{-1}(u_m) \psi_k^{-1}(u_{m-1}) \dots \psi_k^{-1}(u_{j_1+1}) \underbrace{00 \dots 0}_{kj_1})$$

Since we need at most m computations of $\psi_k^{-1}(u_j)$ and n additions, this last step does not increase the overall complexity. ■

Thus Algorithm 3 is an alternative to Algorithm 2, with the same complexity order.

Remark 39. Both, Algorithm 2 and Algorithm 3, can be improved if the input codeword is represented in row reduced echelon form. Then v should not be a random element, but the first row of the input matrix. This implies that u is already normalized, which improves the complexity of both message retrieval algorithms.

V. MESSAGE ENCODING FOR CYCLIC ORBIT CODES

Recall that an orbit code $\mathcal{C} \subseteq \mathcal{G}_q(k, n)$ is defined as the orbit of a given $\mathcal{U} \in \mathcal{G}_q(k, n)$ under the action of a subgroup G of GL_n . In general it holds that $|\mathcal{C}| \leq |G|$, and not necessarily $|\mathcal{C}| = |G|$, i.e., some elements of G might generate the same codewords. Denote by

$$\text{stab}_{\text{GL}_n}(\mathcal{U}) := \{A \in \text{GL}_n \mid \mathcal{U}A = \mathcal{U}\}$$

the stabilizer of \mathcal{U} in GL_n , and by $G/\text{stab}_{\text{GL}_n}(\mathcal{U})$ the set of all right cosets $\text{stab}_{\text{GL}_n}(\mathcal{U})A$ for $A \in \text{GL}_n$. We define the map

$$g : G/\text{stab}_{\text{GL}_n}(\mathcal{U}) \longrightarrow \mathcal{G}_q(k, n) \\ [A] \longmapsto \mathcal{U}A.$$

where $[A]$ denotes the coset of A .

Theorem 40. *The map g is injective.*

Proof: Let $A, B \in G$. Assume that $g(A) = g(B) \iff \mathcal{U}A = \mathcal{U}B$, then

$$AB^{-1} \in \text{stab}_{\text{GL}_n}(\mathcal{U}),$$

and thus $A = AB^{-1}B \in \text{stab}_{\text{GL}_n}(\mathcal{U})B$. Hence, A and B are in the same right cosets of $\text{stab}_{\text{GL}_n}(\mathcal{U})$, which implies the statement. ■

For the remainder of this paper we will restrict ourselves to cyclic orbit codes, since these have simpler message encoders. Moreover, they have more useful structure than other orbit codes and are therefore better understood from a construction and error decoding point of view.

Cyclic orbit codes are those codes that can be defined by the action of a cyclic subgroup G , i.e., $G = \langle P \rangle$ for some matrix $P \in \text{GL}_n$. This notion is not to be mistaken with the definition of cyclic subspace codes from [2], [8], which are unions of special cyclic orbit codes with different initial subspaces. For cyclic orbit codes one clearly has a bijection from $\mathcal{M} = \{0, 1, \dots, \text{ord}(P) - 1\}$ to G , namely

$$\begin{aligned} h' : \quad \{0, 1, \dots, \text{ord}(P) - 1\} &\longrightarrow G \\ i &\longmapsto P^i. \end{aligned}$$

From group theory (see e.g. [19]) one knows that $|G/\text{stab}_{\text{GL}_n}(\mathcal{U})|$ is a divisor of $|G| = \text{ord}(P)$ and that if $\text{ord}_{\mathcal{U}}(P) := |G/\text{stab}_{\text{GL}_n}(\mathcal{U})| < |G|$, then $\mathcal{U}P^i = \mathcal{U}P^{i+\text{ord}_{\mathcal{U}}(P)}$. Thus it follows:

Lemma 41. *The map*

$$\begin{aligned} h : \quad \{0, 1, \dots, \text{ord}_{\mathcal{U}}(P) - 1\} &\longrightarrow G/\text{stab}_{\text{GL}_n}(\mathcal{U}) \\ i &\longmapsto [P^i]. \end{aligned}$$

is a bijection for any $\mathcal{U} \in \mathcal{G}_q(k, n)$.

Corollary 42. *The map $\text{enc}_2 := g \circ h$ is injective and hence is an encoding map from the message set $\mathcal{M} = \{0, \dots, \text{ord}_{\mathcal{U}}(P) - 1\}$ to the cyclic orbit code $\mathcal{C} = \mathcal{U}\langle P \rangle \subseteq \mathcal{G}_q(k, n)$.*

Note that enc_2 can be computed straightforwardly, with matrix multiplications. Its inverse, i.e., the message retrieval map, is based on a discrete logarithm problem (DLP), which is in general known to be a hard problem. There are many results on when the DLP is hard and when it is not; for a survey of various algorithms and their complexities see e.g. [27]. In the following we will investigate some special types of cyclic orbit codes with respect to the computation of enc_2 and enc_2^{-1} .

A. Primitive Cyclic Orbit Codes

For this subsection let α be a primitive element of \mathbb{F}_{q^n} , $p(x) \in \mathbb{F}_q[x]$ its minimal polynomial and P the corresponding companion matrix. Denote by $G = \langle P \rangle$ the group generated by it. Because of the primitivity it holds that

$$\text{ord}(\alpha) = \text{ord}(P) = |G| = q^n - 1.$$

We call $\mathcal{C} = \mathcal{U}G$ a *primitive cyclic orbit code* for any $\mathcal{U} \in \mathcal{G}_q(k, n)$. For more information on the cardinality and minimum distance of different primitive cyclic orbit codes the interested reader is referred to [15], [37]. We can now state the message encoding algorithm for primitive cyclic orbit codes. For this let $U \in \mathbb{F}_q^{k \times n}$ be a matrix, such that $\text{rs}(U) = \mathcal{U}$.

Algorithm 4 Message encoding for a primitive cyclic orbit code $\mathcal{C} = \mathcal{U}\langle P \rangle \subseteq \mathcal{G}_q(k, n)$.

Require: Message $i \in \{0, 1, \dots, \text{ord}_{\mathcal{U}}(P) - 1\}$.

 Compute P^i .

 Compute $V = UP^i$.

return $\mathcal{V} = \text{rs}(V)$

Theorem 43. *The computational complexity of Algorithm 4 is in $\mathcal{O}_q(n^3)$.*

Proof: Since P is a companion matrix of a primitive polynomial, we can compute P^i as described in Lemma 22. Hence, this can be done with $\mathcal{O}_q(n^3)$ operations. The multiplication with $U \in \mathbb{F}_q^{k \times n}$ can be done with kn^2 operations, thus the overall complexity is in $\mathcal{O}_q(n^3)$. ■

For the message retrieval map we assume that an error correcting decoding algorithm has already found P^i such that $\mathcal{U}P^i$ is the respective codeword. This is a realistic assumption, as can be seen in the decoding algorithms of [37]. The retrieval map then needs to solve a discrete logarithm in the group $\langle P \rangle$ to find the exponent i , which corresponds to the message. The group $\langle P \rangle$ has order $q^n - 1$. Since $\mathbb{F}_q[P] \cong \mathbb{F}_q[\alpha]$, we can equivalently compute the discrete logarithm in the group $\langle \alpha \rangle$.

There are several known algorithms to compute discrete logarithms. In this paper we will work with the well-known Pohlig-Hellmann algorithm, see e.g. [26, Sec. 3.6.3].

Lemma 44. [26] *The Pohlig-Hellman algorithm for computing a solution for the discrete logarithm in a group of order $q^n - 1$ has a computational complexity in $\mathcal{O}_{q^n}(\sum_{i=1}^r e_i(\log_2 q^n + \sqrt{p_i}))$, where $\prod_{i=1}^r p_i^{e_i}$ is the prime factorization of $q^n - 1$.*

We describe a message retrieval algorithm for a primitive cyclic orbit code $\mathcal{C} = \mathcal{U}\langle P \rangle \subseteq \mathcal{G}_q(k, n)$ using the Pohlig-Hellman algorithm in Algorithm 5. In the algorithm, $\rho : \mathbb{F}_q[\alpha] \rightarrow \mathbb{F}_q[P]$ denotes the isomorphism introduced in Section II.

Algorithm 5 Message retrieval for a primitive cyclic orbit code $\mathcal{C} = \mathcal{U}\langle P \rangle \subseteq \mathcal{G}_q(k, n)$.

Require: A codeword $\mathcal{V} \in \mathcal{C}$ and P^i such that $\mathcal{U}P^i = \mathcal{V}$.

Compute $\beta = \rho^{-1}(P^i)$.

Use the Pohlig-Hellman algorithm to find $i = \log_\alpha \beta$.

return i

Theorem 45. *Let $\prod_{i=1}^r p_i^{e_i}$ be the prime factorization of $q^n - 1$. The computational complexity of Algorithm 5 is in*

$$\mathcal{O}_q(n^3 \log_2 q \sum_{i=1}^r e_i + n^2 \sum_{i=1}^r e_i \sqrt{p_i})$$

Proof: Since P is the companion matrix of α , β is simply $\phi_{1,n}$ of the first row of P^i , hence, by Lemma 21, the computation of β can be done with at most n operations over \mathbb{F}_q . Then it follows from Lemma 44 that the discrete logarithm can be computed with a complexity in $\mathcal{O}_q(n^3 \log_2 q \sum_{i=1}^r e_i + n^2 \sum_{i=1}^r e_i \sqrt{p_i})$. Since any operation in \mathbb{F}_{q^n} can be done with at most $\mathcal{O}_q(n^2)$ operations over \mathbb{F}_q , the statement follows. ■

We can upper bound this complexity for cyclic orbit codes in $\mathcal{G}_q(k, n)$, in the case where $q^n - 1$ is n^2 -smooth as follows.

Corollary 46. *If $q^n - 1$ is n^2 -smooth (i.e., if all prime factors of $q^n - 1$ are less than or equal to n^2) and all e_i are less than or equal to k , then the complexity order of Algorithm 5 is upper bounded by $\mathcal{O}_q(n^3 k r \log_2 q)$, where r is the number of distinct prime factors of $q^n - 1$.*

From an application point of view a complexity order of at most $\mathcal{O}_q(n^3 k r \log_2 q)$ is reasonable if r is upper bounded by n . If we assume e.g. that $q \leq 2^k$, then we can simplify the above complexity order to $\mathcal{O}_q(n^3 k^2 r)$. For comparison, the complexities of the decoders in [21], [32] are at least cubic in n and the decoding complexity of the rank-based error decoding algorithm for primitive cyclic orbit codes in [37] is of order $\mathcal{O}_q(q^{2k}(n^2 + k^2 n))$.

The following question remains: for which values of q and n is $q^n - 1$ n^2 -smooth? For $q = 2$ and $q = 3$ Tables I and II show values of $n \leq 60$ for which $q^n - 1$ is n^2 -smooth. As explained before, the Pohlig-Hellman algorithm for these cases has a complexity of order at most $\mathcal{O}_q(n^3 k^2 r)$ if k is at least $\max\{e_i \mid i = 1, \dots, r\}$. As one can see, the largest respective exponents e_i for the values presented in Table I are less than or equal to 3, hence k should be at least 3. This is not much of a restriction, since for $k \leq 2$, any constant dimension code in $\mathcal{G}_q(k, n)$ is no-error-correcting. In Table II the values for e_i are larger, hence the restriction on k is stricter. But since we are only considering complexity orders, we can allow e_i to be slightly greater than k , without impairing the overall complexity order. Furthermore, we can see that r is reasonably small for these parameter sets, which is necessary for an efficient performance of the Pohlig-Hellman algorithm.

Remark 47. There are values for q and n , where $q^n - 1$ is not n^2 -smooth but the largest prime factor of $q^n - 1$ is close to n^2 . Then the complexity order of Algorithm 5 will still be $\mathcal{O}_q(n^3 k^2 r)$ and the message retrieval algorithm will thus still not increase the overall decoding complexity for many parameters.

B. Unions of Primitive Cyclic Orbit Codes

First we want to generalize the previously discussed decoding algorithm to unions of primitive cyclic orbit codes. As seen in [8], [15], [20], unions of primitive cyclic orbit codes are among the best known constructions for constant dimension codes. Decoding such codes can be done by using a decoding algorithm for single orbits for each of the orbits that constitute the code. In the error correction decoding process the algorithm needs to decide which orbit the respective closest codeword is on. This information can then be passed on to the message retrieval algorithm, which simply applies Algorithm 5 on that chosen orbit.

As before let $\alpha \in \mathbb{F}_{q^n}$ be a primitive element and P its companion matrix. The various orbits that form our code as a union are all generated by the action of P (respectively α). They are given by the initial points $\mathcal{U}_1, \dots, \mathcal{U}_z \in \mathcal{G}_q(k, n)$. For simplicity we assume that all orbits have the same cardinality c^* . Then the following map is an encoding map for the code

n	$\max p_i$	$\max e_i$	$\max(e_i n, e_i p_i)$	r	n^2
6	7	2	18	2	36
8	17	1	17	3	64
9	73	1	73	2	81
10	31	1	31	3	100
11	89	1	89	2	121
12	13	2	24	4	144
14	127	1	127	3	196
15	151	1	151	3	225
18	73	3	73	4	324
20	41	2	41	5	400
21	337	2	337	3	441
24	241	2	241	6	576
28	127	1	127	6	784
30	331	2	331	6	900
36	109	3	109	8	1296
48	673	2	673	9	2304
60	1321	2	1321	11	3600

TABLE I
VALUES OF $n \leq 60$ FOR WHICH $2^n - 1 = \prod_{i=1}^r p_i^{e_i}$ IS n^2 -SMOOTH.

n	$\max p_i$	$\max e_i$	$\max(e_i n, e_i p_i)$	r	n^2
6	13	3	18	3	36
8	41	5	41	3	64
10	61	3	61	3	100
12	73	4	73	5	144
16	193	6	193	5	256

TABLE II
VALUES OF $n \leq 60$ FOR WHICH $3^n - 1 = \prod_{i=1}^r p_i^{e_i}$ IS n^2 -SMOOTH.

$$\mathcal{C} = \bigcup_{i=1}^z \mathcal{U}_i \langle P \rangle \subseteq \mathcal{G}_q(k, n):$$

$$\text{enc}_3 : \{0, \dots, zc^* - 1\} \longrightarrow \mathcal{G}_q(k, n)$$

$$i \longmapsto \mathcal{U}_j P^i, \quad j = \left\lceil \frac{i+1}{c^*} \right\rceil$$

Note that, because of the cardinality of the orbits, $\mathcal{U}_j P^i = \mathcal{U}_j P^{i+c^*}$. Therefore we can also compute P^ℓ , where $\ell \equiv i \pmod{c^*}$, instead of P^i in the above encoding map. For the inverse map, i.e., the message retrieval, we require the information j (which orbit the word is on) from the error decoder. Then we use Algorithm 5 to retrieve the respective exponent ℓ of P . With both these pieces of information we can easily reconstruct the message $i = \ell + (j-1)c^*$.

Example 48. Let $q = 2, n = 4, k = 2$ and $\alpha \in \mathbb{F}_{2^4}$ be a root of the irreducible polynomial $x^4 + x + 1$ and denote by P its companion matrix. Moreover, $\rho : \mathbb{F}_2[\alpha] \rightarrow \mathbb{F}_2[P]$ is the isomorphism, as previously described. One has $\text{ord}(\alpha) = 15$, i.e., α is primitive. Let the extension field representations of the initial points $\mathcal{U}_1, \mathcal{U}_2 \in \mathcal{G}_2(2, 4)$ be $\{0, 1, \alpha, \alpha+1\}$ and $\{0, 1, \alpha^2, \alpha^2+1\}$, respectively. Both orbits, $\mathcal{U}_1 \langle P \rangle$ and $\mathcal{U}_2 \langle P \rangle$, have cardinality $c^* = 15$. Our code consists of the union of these two orbits, i.e., our message set is $\mathcal{M} = \{0, \dots, 29\}$. Assume we received some $\mathcal{R} \in \mathcal{G}_2(2, 4)$ that is error decoded to the codeword $\mathcal{U}_2 \rho(\beta)$ with $\beta = \alpha^3 + \alpha + 1$. We use Algorithm 5 to compute $\ell = \log_\alpha \beta = 7$ and retrieve the message $i = \ell + (j-1)c^* = 7 + 15 = 22$.

C. Non-Primitive Irreducible Cyclic Orbit Codes

The second generalization that we want to consider is the message retrieval of cyclic orbit codes $\mathcal{U} \langle P \rangle \subseteq \mathcal{G}_q(k, n)$ that are not generated by a companion matrix of a primitive polynomial, but rather of a non-primitive irreducible one. For more information on the distinctions of primitive, irreducible non-primitive and completely reducible cyclic orbit codes the interested reader is referred to [15], [37]. For the irreducible (but not primitive) case we assume that $\alpha \in \mathbb{F}_{q^n}$ is irreducible of order less than (and a divisor of) $q^n - 1$. Then $\mathbb{F}_{q^n}^*$ is partitioned into different orbits under the action of α . The various vectors of a codeword possibly lie on several of these orbits, which the error correction algorithm needs to take into consideration. The message retrieval algorithm then only needs to compute the discrete logarithm in $\langle \alpha \rangle$, just as in the primitive case. The computation of the discrete logarithm is possibly easier than in the primitive case, since the group order is smaller.

Example 49. Let $q = 2, n = 4, k = 2$ and $\alpha \in \mathbb{F}_{2^4}$ be a root of the irreducible polynomial $x^4 + x^3 + x^2 + x + 1$. As before, $\rho : \mathbb{F}_2[\alpha] \rightarrow \mathbb{F}_2[P]$ denotes an isomorphism. We have $\text{ord}(\alpha) = 5$, thus our message set is $\mathcal{M} = \{0, \dots, 4\}$. $\mathbb{F}_{2^4}^*$ is partitioned into the orbits $\langle \alpha \rangle, (\alpha+1)\langle \alpha \rangle$ and $(\alpha^2+1)\langle \alpha \rangle$. Let $\mathcal{U} \in \mathcal{G}_2(2, 4)$, P be the companion matrix of α and $\mathcal{C} = \mathcal{U} \langle P \rangle$ be the

cyclic orbit we want to consider. Assume that we received some word $\mathcal{R} \in \mathcal{G}_2(2, 4)$, which was error decoded to the codeword $\mathcal{U}\rho(\beta)$ with $\beta = \alpha^3 + \alpha^2 + \alpha + 1$. We use Algorithm 5 to get the message $i = \log_\alpha \beta = 4$.

In the next example we illustrate two cases where the order of a primitive element is not n^2 -smooth, but the order of some non-primitive irreducible element is n^2 -smooth. This shows that there are more irreducible cyclic orbit codes with an efficiently computable message retrieval map than only the primitive ones.

Example 50. 1) Let $q = 3, n = 18, k = 9$. Then any primitive element of $\mathbb{F}_{3^{18}}$ has order $3^{18} - 1 = 2^3 \cdot 7 \cdot 13 \cdot 19 \cdot 37 \cdot 757$, which is not n^2 -smooth. But there also exists an irreducible element in $\mathbb{F}_{3^{18}}$ of order $(3^{18} - 1)/(3^9 - 1) = 2^2 \cdot 7 \cdot 19 \cdot 37$, which is n^2 -smooth.
2) Let $q = 3, n = 48, k = 24$. Then any primitive element of $\mathbb{F}_{3^{48}}$ has order $3^{48} - 1$, whose largest prime power is 6481. Hence this order is not n^2 -smooth. But there also exists an irreducible element in $\mathbb{F}_{3^{48}}$ of order $(3^{48} - 1)/(3^{24} - 1) = 2 \cdot 17 \cdot 97 \cdot 193 \cdot 577 \cdot 769$, which is n^2 -smooth.

D. Completely Reducible Cyclic Orbit Codes

Lastly we briefly explain how the previous results can be generalized to completely reducible cyclic orbit codes. For these codes the generating matrix P is not a companion matrix of some irreducible polynomial, but rather a completely reducible matrix. I.e., P can be brought into block diagonal form, where each block is again a companion matrix of some irreducible polynomial. For simplicity we assume that $P \in \mathbb{F}_q^{n \times n}$ is of the form

$$P = \begin{pmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_t \end{pmatrix},$$

where $P_i \in \mathbb{F}_q^{n_i \times n_i}$ is a companion matrix of a primitive element in $\mathbb{F}_{q^{n_i}}$. One can easily see that $n = n_1 + n_2 + \dots + n_t$. Then any element $u \in \mathbb{F}_q^n$ can be represented as an element of $\mathbb{F}_{q^{n_1}} \times \mathbb{F}_{q^{n_2}} \times \dots \times \mathbb{F}_{q^{n_t}}$. Furthermore, we can represent any element of the code this way, thus each of these blocks of length n_i can be seen as a primitive cyclic orbit code. For a more detailed explanation of completely reducible orbit codes see [37]. The message encoding and retrieval can now be done in each of the blocks of length n_i . For the retrieval one gets t integer solutions i_1, \dots, i_t . The final solution i , such that $\mathcal{U}P^i$ is the received word, is then given by solving the system of simultaneous congruences $i \equiv i_j \pmod{n_j}$ for $j = 1, \dots, t$.

Remark 51. The linkage construction for cyclic orbit codes from [15] is closely related to unions of completely reducible orbit codes, see [15, Proposition 5.3]. The message retrieval map explained before can be extended to work for the linkage construction as well.

VI. A HYBRID ENCODER FOR SEMI-LINEARLY ISOMETRIC CODES

We have seen in the previous section that there exist parameters for which enc_2 is a message encoding function for orbit codes, that has an efficient inverse map, i.e., an efficient corresponding retrieval map. For many parameters though, the procedures described in Section V are not efficiently computable. In this section we show how semi-linear isometry can be useful for message encoding and retrieval purposes. We will describe the results in general, for any pair of semi-linearly isometric codes, and then explain how this can be applied to the special class of spreads constructed as primitive cyclic orbit codes, since these are always (semi-linearly isometric to) Desarguesian spreads (cf. e.g. [1, Theorem 14]).

Let $A \in \text{GL}_n$ and $\sigma \in \text{Aut}(\mathbb{F}_q)$ a field automorphism. If $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{G}_q(k, n)$ are two constant dimension codes, such that $\sigma(\mathcal{C}_1 A) = \mathcal{C}_2$ (as sets of vector spaces), where σ is applied element-wise on the codewords of $\mathcal{C}_1 A$, then we call \mathcal{C}_1 and \mathcal{C}_2 *semi-linearly isometric*. If $\sigma = \text{id}$, then we call the codes *linearly isometric*. This name *isometric* arises because the codes have the same cardinality and distance distribution. For more information on semi-linear isometry of subspace codes the interested reader is referred to [34], [35].

Assume that there exists an encoder enc for the code \mathcal{C}_1 (for a message set \mathcal{M}). Then

$$\begin{aligned} \text{enc}' : \mathcal{M} &\longrightarrow \mathcal{G}_q(k, n) \\ i &\longmapsto \sigma(\text{enc}(i)A) \end{aligned}$$

is an encoder for \mathcal{C}_2 . We call this a *hybrid encoder* for \mathcal{C}_1 and \mathcal{C}_2 .

Theorem 52. Let enc and enc' be as above. Denote the complexity order of enc by $\omega(\text{enc})$. Then the complexity of the hybrid encoder enc' is in $\mathcal{O}_q(\omega(\text{enc}) + kn^2)$.

Proof: Follows straightforwardly from the fact that the computational complexity order of the multiplication with A is in $\mathcal{O}_q(kn^2)$. The complexity of the field automorphism is negligible. \blacksquare

As an example we want to show how the idea of a hybrid encoder can be used for Desarguesian spread codes. As mentioned above, any spread constructed as a primitive cyclic orbit code is a Desarguesian spread, in the more general definition of Desarguesian spread. It follows that such a primitive cyclic orbit code is semi-linearly isometric to a code from Construction I (see also [1, Corollary 16]). With this knowledge we can use enc_1 to efficiently encode and retrieve messages, but use the orbit code structure for an error correcting decoding algorithm, e.g. the coset leader decoding algorithm from [37].

Example 53. Let \mathcal{C}_1 be the spread constructed in Example 13 and let \mathcal{C}_2 be the orbit spread code constructed in Example 16, both subsets of $\mathcal{G}_2(2, 4)$ with five elements. Then we can use the algorithm of Feulner from [9]¹ to find a linear transformation $A \in \text{GL}_4$, such that $\mathcal{C}_1 A = \mathcal{C}_2$. One such linear transformation is given by

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

Let β be a primitive element of \mathbb{F}_{2^4} . In the isomorphic extension field representation, A maps the basis $\{1, \beta, \beta^2, \beta^3\}$ of \mathbb{F}_{2^4} over \mathbb{F}_2 to the new basis $\{1, \beta^2 + \beta, \beta + 1, \beta^3 + \beta\}$. We can now use \mathcal{C}_1 for message encoding, say we encode a given message to the codeword $\mathcal{U} = \psi_4^{-1}\{0, 1, \beta, \beta + 1\} \in \mathcal{C}_1$, then we send $\mathcal{U}A = \psi_4^{-1}\{0, 1, \beta^2 + \beta, \beta^2 + \beta + 1\} \in \mathcal{C}_2$ over the channel. We can then do error correction decoding in the code \mathcal{C}_2 with e.g. the coset leader decoder. Say we decoded the received word to the sent codeword $\mathcal{U}A \in \mathcal{C}_2$. Then it suffices to apply A^{-1} on only one of the non-zero elements of $\mathcal{U}A$, e.g. $\psi_4^{-1}(\beta^2 + \beta)A^{-1} = \psi_4^{-1}(\beta) = (0, 1, 0, 0)$ to identify the corresponding codeword $\mathcal{U} \in \mathcal{C}_1$, from which we can then easily get the message as explained in Section IV.

VII. CONCLUSION

In this work we investigate how message encoding can be done for spread and cyclic orbit codes, two families of subspace codes that have been well-studied for error correction in random network coding.

We show that for Desarguesian spread codes in $\mathcal{G}_q(k, n)$ there exists an encoding map such that the map itself and the inverse map are efficiently computable with a computational complexity of order at most $\mathcal{O}_q(kn)$. In addition, we study the method of enumerative coding for this family of codes and show that the first message retrieval map, with a little twist, is equal to the indexing function of enumerative coding.

Furthermore, we develop an encoder for general cyclic orbit codes. This map is efficiently computable, but the inverse, i.e., the message retrieval map, involves the computation of a discrete logarithm. This is known to be computationally hard in general, but we show for which parameters the Pohlig-Hellman algorithm computes the discrete logarithm in complexity of order at most $\mathcal{O}_q(n^3 k r \log_2 q)$. This is done in detail for primitive cyclic orbit codes. Moreover, some remarks on how to generalize these results to unions of cyclic orbit codes and completely reducible cyclic orbit codes are given.

In the end we propose a hybrid encoder for semi-linearly isometric codes, which is useful if one knows an efficient message retrieval map for a linearly isometric code to a given one. We show how this can be realized for cyclic orbit codes that are Desarguesian spreads, such that one can use the orbit structure for error correction, but avoid the discrete logarithm problem in the message retrieval part.

An open question for further research is, if there are other, for certain parameter sets more efficient, ways to solve the discrete logarithm problem in the message retrieval of orbit codes. Moreover, it would be interesting to find other families of semi-linearly isometric codes where a hybrid encoder can be helpful to combine efficient error correction decoders with efficient message retrieval maps.

ACKNOWLEDGMENT

The author would like to thank Yuval Cassuto for his reference to enumerative coding, John Sheekey for his advice on Desarguesian spreads, and Margreta Kuijper for fruitful discussions and comments on this work. She would furthermore like to thank the anonymous reviewers for their valuable comments.

REFERENCES

- [1] L. Bader and G. Lunardon. Desarguesian spreads. *Ricerche di Matematica*, 60(1):15–37, 2011.
- [2] E. Ben-Sasson, T. Etzion, A. Gabizon, and N. Raviv. Subspace polynomials and cyclic subspace codes. *arXiv:1404.7739 [cs.IT]*, 2014.
- [3] M. Bossert and E.M. Gabidulin. One family of algebraic codes for network coding. In *Proceedings of the 2009 IEEE International Symposium on Information Theory (ISIT)*, pages 2863–2866, 2009.
- [4] T. M. Cover. Enumerative source encoding. *IEEE Transactions on Information Theory*, 19(1):73–77, Jan 1973.
- [5] P. Delsarte. An algebraic approach to the association schemes of coding theory. *Philips Journal of Research*, (10):vi+97, 1973.
- [6] T. Etzion and N. Silberstein. Error-correcting codes in projective spaces via rank-metric codes and Ferrers diagrams. *IEEE Transactions on Information Theory*, 55(7):2909–2919, March 2009.

¹This algorithm requires two codes in $\mathcal{G}_q(k, n)$ as input and then computes if they are linearly isometric; and if so, finds the linear transformation from one code into the other.

- [7] T. Etzion and N. Silberstein. Codes and designs related to lifted MRD codes. *IEEE Transactions on Information Theory*, 59(2):1004–1017, 2013.
- [8] T. Etzion and A. Vardy. Error-correcting codes in projective space. *IEEE Transactions on Information Theory*, 57(2):1165–1173, 2011.
- [9] T. Feulner. Canonical forms and automorphisms in the projective space. *arXiv:1305.1193 [cs.IT]*, 2013.
- [10] E. M. Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985.
- [11] E. M. Gabidulin and N. I. Pilipchuk. Multicomponent network coding. In *Proceedings of the Seventh International Workshop on Coding and Cryptography (WCC) 2011*, pages 443–452, Paris, France, 2011.
- [12] M. Gadouleau and Z. Yan. Constant-rank codes and their connection to constant-dimension codes. *IEEE Transactions on Information Theory*, 56(7):3207–3216, 2010.
- [13] J. von zur Gathen. Efficient and optimal exponentiation in finite fields. *Comput. Complexity*, 1(4):360–394, 1991.
- [14] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, Cambridge, second edition, 2003.
- [15] H. Gluesing-Luerssen, K. Morrison, and C. Troha. Cyclic orbit codes and stabilizer subfields. *arXiv:1403.1218 [cs.IT]*, 2014.
- [16] E. Gorla, F. Manganiello, and J. Rosenthal. An algebraic approach for decoding spread codes. *Advances in Mathematics of Communications (AMC)*, 6(4):443 – 466, 2012.
- [17] E. Gorla and A. Ravagnani. Partial spreads in random network coding. *Finite Fields and Applications*, 26:104–115, 2014.
- [18] J. W. P. Hirschfeld. *Projective Geometries over Finite Fields*. Oxford Mathematical Monographs. The Clarendon Press Oxford University Press, New York, second edition, 1998.
- [19] A. Kerber. *Applied finite group actions*, volume 19 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1999.
- [20] A. Kohnert and S. Kurz. Construction of large constant dimension codes with a prescribed minimum distance. In J. Calmet, W. Geiselmann, and J. Müller-Quade, editors, *MMICS*, volume 5393 of *Lecture Notes in Computer Science*, pages 31–42. Springer, 2008.
- [21] R. Kötter and F. R. Kschischang. Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory*, 54(8):3579–3591, 2008.
- [22] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge University Press, Cambridge, London, 1994. Revised edition.
- [23] F. Manganiello, E. Gorla, and J. Rosenthal. Spread codes and spread decoding in network coding. In *Proceedings of the 2008 IEEE International Symposium on Information Theory (ISIT)*, pages 851–855, Toronto, Canada, 2008.
- [24] F. Manganiello and A.-L. Trautmann. Spread decoding in extension fields. *Finite Fields and Applications*, 25:94–105, jan 2014.
- [25] F. Manganiello, A.-L. Trautmann, and J. Rosenthal. On conjugacy classes of subgroups of the general linear group and cyclic orbit codes. In *Proceedings of the 2011 IEEE International Symposium on Information Theory (ISIT)*, pages 1916–1920, St. Petersburg, Russia, 2011.
- [26] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC Press Series on Discrete Mathematics and its Applications. CRC Press, Boca Raton, FL, 1997. With a foreword by Ronald L. Rivest.
- [27] A. M. Odlyzko. Discrete logarithms in finite fields and their cryptographic significance. In Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, editors, *Advances in Cryptology*, volume 209 of *Lecture Notes in Computer Science*, pages 224–314. Springer Berlin Heidelberg, 1985.
- [28] J. Rosenthal and A.-L. Trautmann. A complete characterization of irreducible cyclic orbit codes and their Plücker embedding. *Designs, Codes and Cryptography*, 66:275–289, 2013.
- [29] M. Schwartz. Gray codes and enumerative coding for vector spaces. *IEEE Transactions on Information Theory*, 60(1):271–281, Jan 2014.
- [30] N. Silberstein and T. Etzion. Enumerative coding for Grassmannian space. *IEEE Transactions on Information Theory*, 57(1):365–374, Jan 2011.
- [31] N. Silberstein and A.-L. Trautmann. Subspace codes based on graph matchings, ferrers diagrams, and pending blocks. *Information Theory, IEEE Transactions on*, 61(7):3937–3953, July 2015.
- [32] D. Silva, F. R. Kschischang, and R. Kötter. A rank-metric approach to error control in random network coding. *IEEE Transactions on Information Theory*, 54(9):3951 –3967, 2008.
- [33] V. Skachek. Recursive code construction for random networks. *IEEE Transactions on Information Theory*, 56(3):1378–1382, 2010.
- [34] A.-L. Trautmann. *Constructions, Decoding and Automorphisms of Subspace Codes*. PhD thesis, University of Zurich, Switzerland, 2013.
- [35] A.-L. Trautmann. Isometry and automorphisms of constant dimension codes. *Advances in Mathematics of Communications (AMC)*, 7(2):147–160, 2013.
- [36] A.-L. Trautmann. Message encoding for spread and orbit codes. In *Proceedings of the 2014 IEEE International Symposium on Information Theory (ISIT)*, pages 2594–2598, 2014.
- [37] A.-L. Trautmann, F. Manganiello, M. Braun, and J. Rosenthal. Cyclic orbit codes. *IEEE Transactions on Information Theory*, 59(11):7386–7404, 2013.
- [38] A.-L. Trautmann, F. Manganiello, and J. Rosenthal. Orbit codes - a new concept in the area of network coding. In *IEEE Information Theory Workshop (ITW)*, pages 1–4, Dublin, Ireland, 2010.
- [39] A.-L. Trautmann and J. Rosenthal. New improvements on the echelon-Ferrers construction. In *Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems – MTNS*, pages 405–408, Budapest, Hungary, 2010.